**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD NUMBER:** 219

**ISSUE TITLE:** AMI Parameter Root Name Clarifications

**REQUESTOR:**  Michael Mirmak, Intel Corp.**DATE SUBMITTED:** March 29, 2022

**DATE REVISED:**

**DATE ACCEPTED:**

**DEFINITION OF THE ISSUE:**

In IBIS 7.1, the structure of the executable AMI\_parameters\_out string is explicitly stated to follow the structure of the AMI\_parameters\_in string, which is provided. The name of the AMI\_parameters\_out string tree varies in the IBIS 7.1 document. From Section 10.2.3, the "root name" is required and is implied to be the first string output; this string is also called the "top level parameter string" and is considered a "parameter group". Section 10.3 describes the .ami file structure and calls the first string the "root name".

Some EDA tools expect and issue warnings enforcing that the parameter tree "root name" of a .ami file must match the first string or "root name" provided by the associated AMI executable (.dll or .so) file, in its AMI\_parameters\_out string. This expectation is apparently based on the language of the specification, but no explicit statement is made that the root name of the .ami file parameter tree and the initial string of the AMI\_parameters\_out and AMI\_parameters\_in strings should match.

The language and the expectations regarding parameter tree root name definitions should be clarified.

**SOLUTION REQUIREMENTS:**

The IBIS specification must meet these requirements:

Table 1: Solution Requirements

|  |  |
| --- | --- |
| Requirement | Notes |
| 1. “Root name” should be clarified, upon all uses in context, to refer to the AMI parameter tree initial string
 | Note that the phrase “root name” is also used in regard to files and directories early in the specification. |
| 1. The specification should clearly state that the AMI parameter definition file root name and the initial identifying string (parameter group name) generated by the AMI executable for the AMI\_parameters\_in and AMI\_parameters\_out strings, if present, should match.
 |  |

**PROPOSED CHANGES:**

In IBIS 7.1, modify the text as follows (changes highlighted in blue). All page numbers refer to the PDF version of the document.

On page 223, the AMI\_parameters\_in text now reads in part:

The AMI\_parameters\_in argument must always be present in the AMI\_Init function call and it must always contain the address of a valid string. The string must always contain at least the root name of the parameter tree, even if there are no parameters to pass to the algorithmic model.

This should be changed to:

The AMI\_parameters\_in argument must always be present in the AMI\_Init function call and it must always contain the address of a valid string. The string must always contain at least the root name of the parameter tree from the corresponding AMI parameter definition file, even if there are no parameters to pass to the algorithmic model. This root name shall be compared to the built-in root name of the receiving algorithmic model by the receiving algorithmic model. The model shall report mismatches as part of its message string (msg), but is not required to return Return Value 0 upon exiting.

All requirements documented above for the formatting of parameters of Usage In for the AMI\_parameters\_in string also apply to parameters defined as Usage InOut when used in the AMI\_parameters\_in string.

On page 224, the parameter string syntax now reads in part:

7. The top level parameter string must be a parameter group.

This should be changed to:

7. The top-level parameter string ~~must~~ shall be the root name and shall be a parameter group (not a parameter value or a parameter name/value pair).

On page 225, the AMI\_parameters\_out text now reads in part:

While the AMI\_parameters\_out argument must always be present in the AMI\_Init function call and the EDA tool must always provide a valid (non-zero) address value in it, algorithmic models are not required to return anything at that address to the EDA tool. For this reason, the EDA tool must also initialize the memory content at that address to zero (null pointer) prior to calling the AMI\_Init function, so that after the execution of the function it can determine whether or not the function returned a valid string pointer at that address. If the AMI\_Init function does not have any parameters to return to the EDA tool, it must return a pointer at the address provided in this argument to a string which contains nothing but the root name. Note that the root name must always be included in the string.

This should be changed to:

While the AMI\_parameters\_out argument must always be present in the AMI\_Init function call and the EDA tool must always provide a valid (non-zero) address value in it, algorithmic models are not required to return anything at that address to the EDA tool. For this reason, the EDA tool must also initialize the memory content at that address to zero (null pointer) prior to calling the AMI\_Init function, so that after the execution of the function it can determine whether or not the function returned a valid string pointer at that address. If the AMI\_Init function does not have any parameters to return to the EDA tool, it ~~must~~ shall return a pointer at the address provided in this argument to a string which contains nothing but the root name. The root name shall be the same root name contained within the executable model. Note that the root name ~~must~~ shall always be included in the string, regardless of any other contents of that string. This executable model root name shall match the root name of the parameter tree from the corresponding AMI parameter definition file. The two root names must be compared by the EDA tool. The EDA tool must report any root name mismatch detected, but may choose to continue or stop simulation at this point.

All requirements documented above for the formatting of parameters of Usage Out for the AMI\_parameters\_out string also apply to parameters defined as Usage InOut when used in the AMI\_parameters\_out string.

Page 233, Section 10.3.2 now reads in part:

The AMI parameter definition file is organized as a “tree”, with “leaves” and “branches” off a single “root” identified by a unique name. A branch may contain an AMI parameter, which itself contains individual leaves, describing features of the model. Branches, unless otherwise noted, may themselves be used to group other branches.

The file shall contain a distinct section or branch named “Reserved\_Parameters” beginning and ending with parentheses. The file may also contain another section or branch named “Model\_Specific”, beginning and ending with parentheses. “Reserved\_Parameters” and “Model\_Specific” are the only branches permitted to be connected to the root of the tree.

The AMI parameter definition file shall be organized in the following way:

This should be changed to:

The AMI parameter definition file is organized as a “tree”, with “leaves” and “branches” off a single “root” identified by a unique root name. A branch may contain an AMI parameter, which itself contains individual leaves, describing features of the model. Branches, unless otherwise noted, may themselves be used to group other branches.

The file shall contain a distinct section or branch named “Reserved\_Parameters” beginning and ending with parentheses. The file may also contain another section or branch named “Model\_Specific”, beginning and ending with parentheses. “Reserved\_Parameters” and “Model\_Specific” are the only branches permitted to be connected to the root of the tree.

The AMI parameter definition file shall be organized ~~in the following way:~~ as shown in the structure below. Note that the first string of non-blank, non-comment characters of the AMI parameter definition file, excluding the opening parenthesis, is the root name.

**BACKGROUND INFORMATION/HISTORY:**

Thanks to Walter Katz of MathWorks for research and suggestions leading to this BIRD.

This BIRD is issued in response to, and uses text from, an IBISCHK7 bug report, number 227: <https://ibis.org/bugs/ibischk/bug227.txt>

Note that not every instance of “must” has been changed to “shall” in the updated text. In general, “shall” is reserved for use with model requirements, which are under the control of the specification; “must” is reserved for use with EDA tool behavior, which is not necessarily controlled by the specification.Draft 2 makes a minor change to the description of the root name in the .ami file as suggested by Mike LaBonte of MathWorks.

Draft 3 captures input from Arpad Muranyi and Randy Wolff, and clarifies the checking of root name matches between EDA tools and executable models, plus InOut parameter rules as requested in the IBIS-ATM Task Group.

Draft 4 includes changes suggested by Arpad Muranyi on the IBIS-ATM reflector on February 28, 2022 as well as during the IBIS-ATM Task Group meeting of March 8, 2022.

Highlighting has been updated in draft 4 to capture all the changes from the original IBIS 7.1 text.